Data Encryption Standard (DES)

Dr. Bhaskar Mondal

DES

- Data Encryption Standard (NBS77)
- Adopted by US Fedral Standards in 1977

History of Data Encryption Standard (DES)

- 1967: Feistel at IBM – Lucifer: block size 128; key size 128 bit
- 1972: NBS asks for an encryption standard
- 1975: IBM developed DES (modification of Lucifer) – block size 64 bits; key size 56 bits
- 1975: NSA suggests modifications
- 1977: NBS adopts DES as encryption standard in (FIPS 46-1, 46-2).
- 2001: NIST adopts Rijndael as replacement to DES.

DES (overview)

- Symmetric Algorithm
- Block Cipher
- Uses a combination of **Substitution and Transpositions (permutations)**
- Called a Product Cipher
- Goes through 16 cycles
- PlainText is organized into 64-bit Blocks
- Uses a 56-bit Key (in reality, 64 bits, but 8 are used as parity-check bits for error control)

DES a Permutation-substitution based cipher

- Initial **Permutation** on Input Text (64-bit)
- Split into Right and Left Halves (32-bit)
- Take right half and **permute** it (Expansion Permutation) 48-bit
- Work on Key (shift) 56-bit, then permute key (48-bits)
- XOR resulting key with right half …result is 32-bit (**S-BoX**)
- Permute result
- XOR result with Left Half
- End of Cycle

DES (cont.)

The next cycle begins with:

- The result of previous cycle as its right half
- The old Right half (48-bit) as Its left half

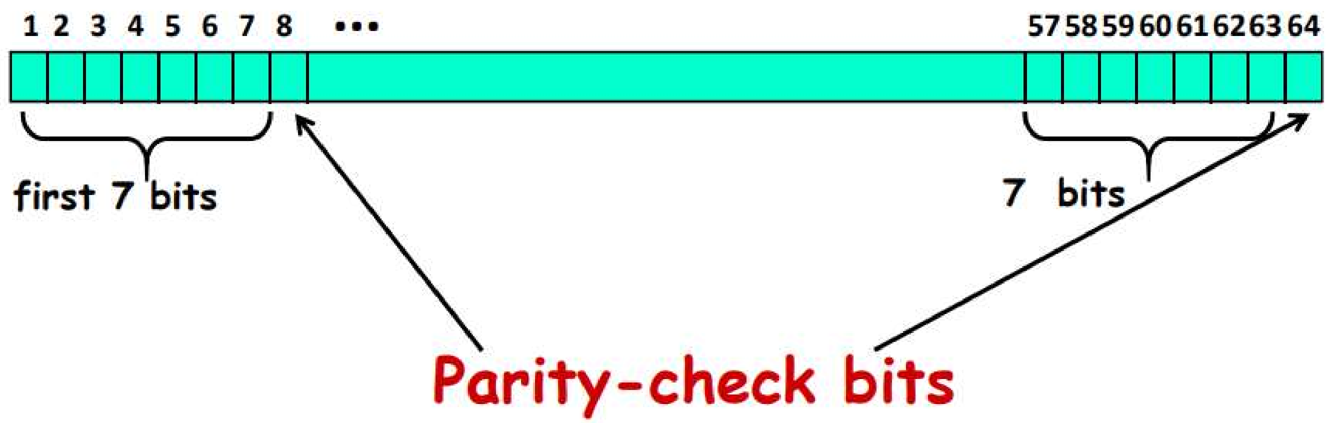
Repeat

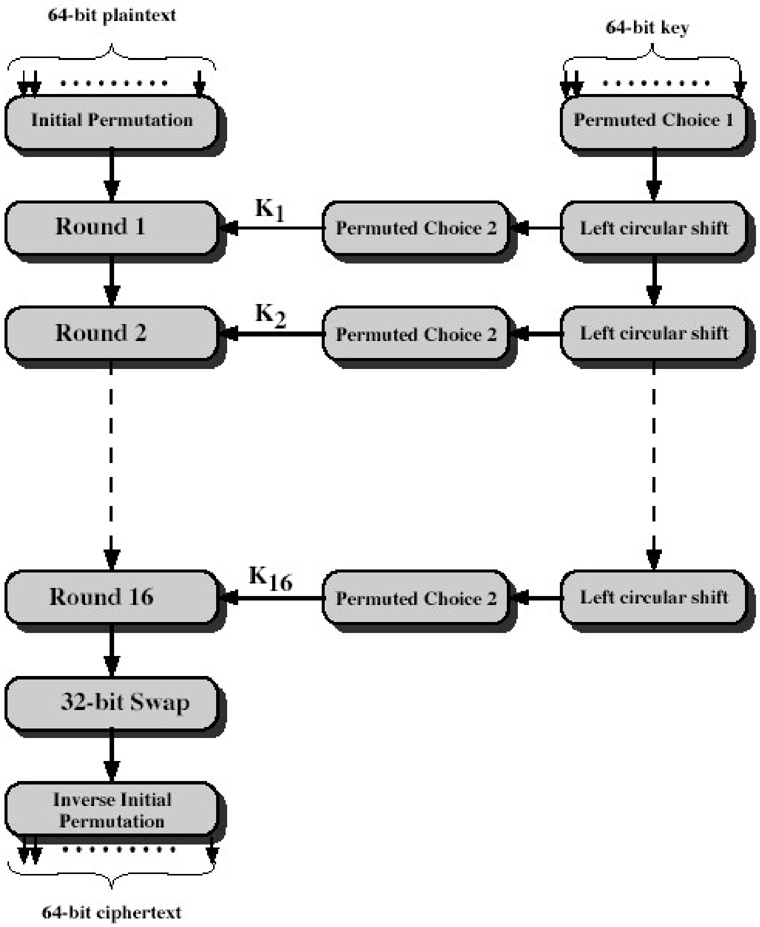
Key Transformation

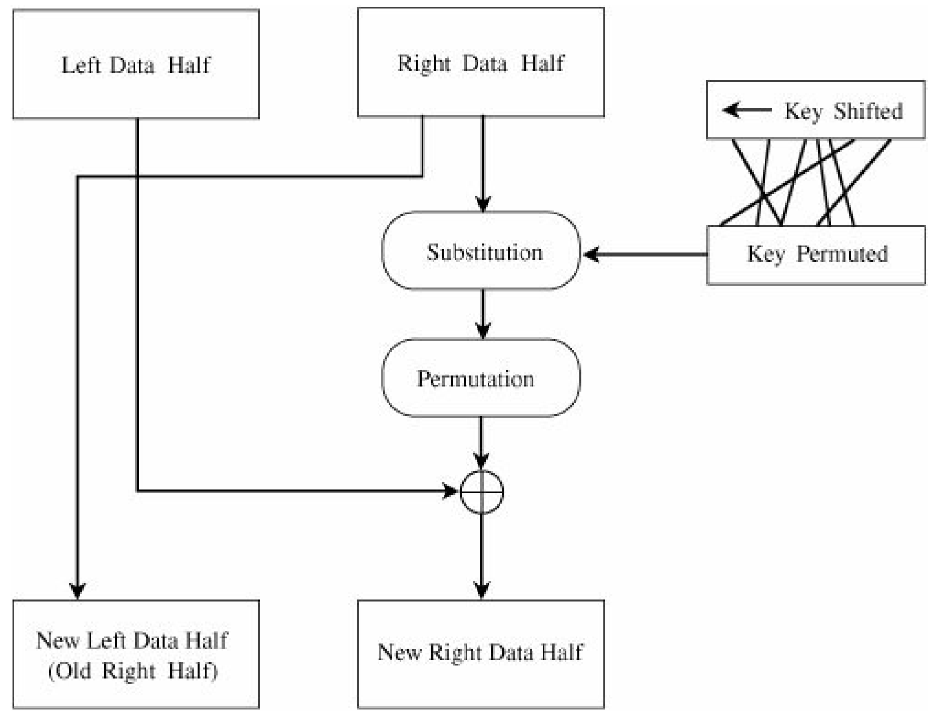
- Starts with 64-bit (16 hexadecimal digits)
- Drop every eighth bit = 56 bits
- Split into two 28-bits halves
- Shift each key to the lift (number of bits)
- Paste both halves
- 48-bit key is then permuted

Key length in DES

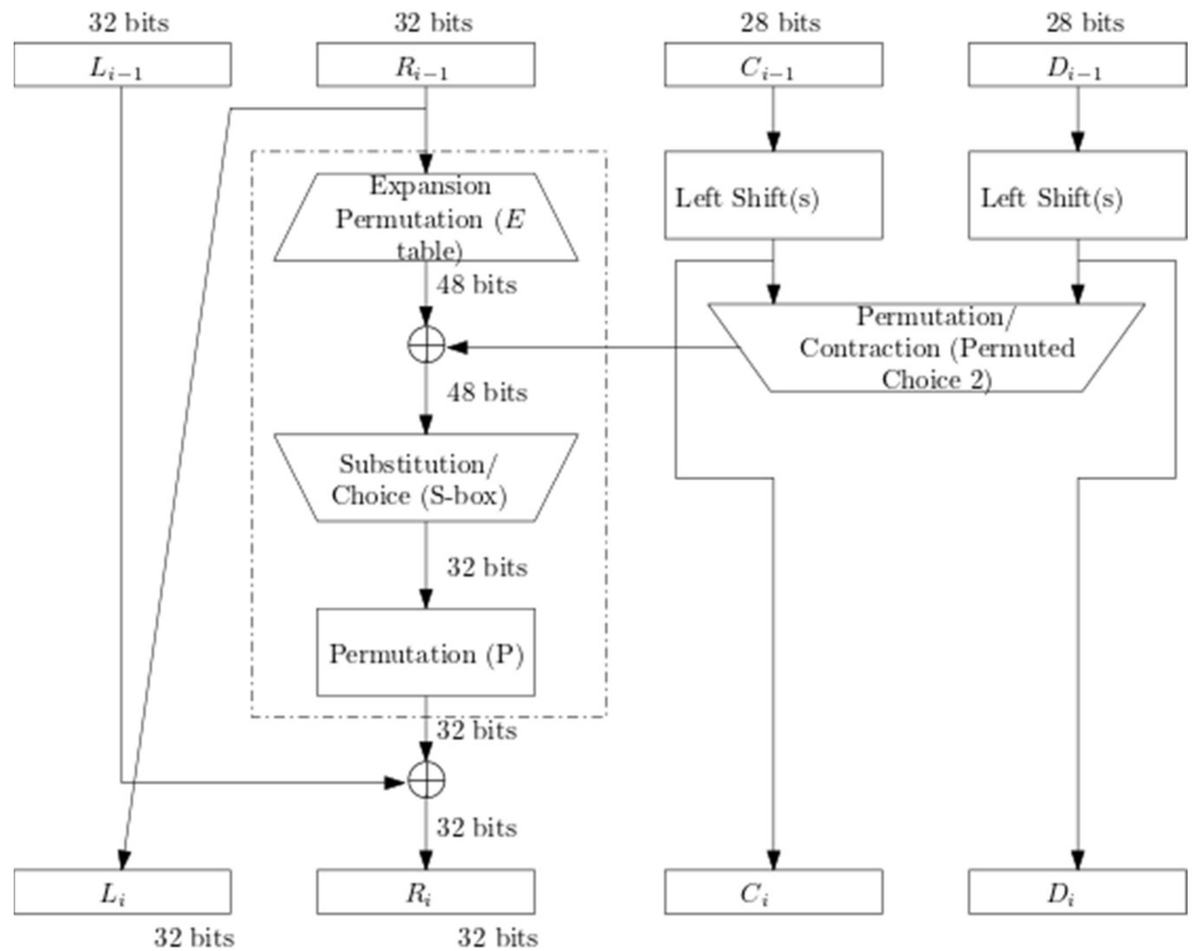
- In the DES specification, the key length is 64 bit:
- 8 bytes; in each byte, the 8th bit is a parity-check bit
- Each parity-check bit is the XOR of the previous 7 bits



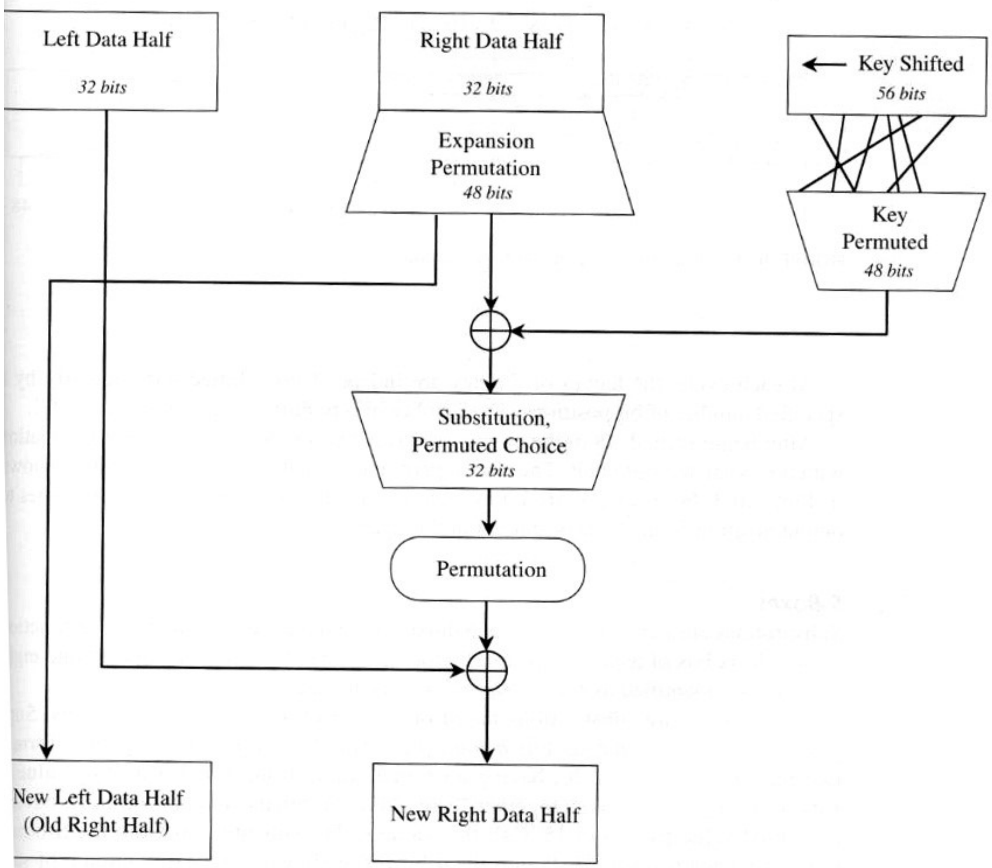

64-bit plaintext
Initial Permutation
Round 1
Round 2
Round 16
32-bit Swap
Inverse Initial Permutation
64-bit ciphertext
64-bit key
Permuted Choice 1
Left circular shift
Left circular shift
Left circular shift
Permuted Choice 2
Permuted Choice 2
Permuted Choice 2
K1
K2
K16


Left Data Half
Right Data Half
Key Shifted
Substitution
Key Permuted
Permutation
New Left Data Half (Old Right Half)
New Right Data Half

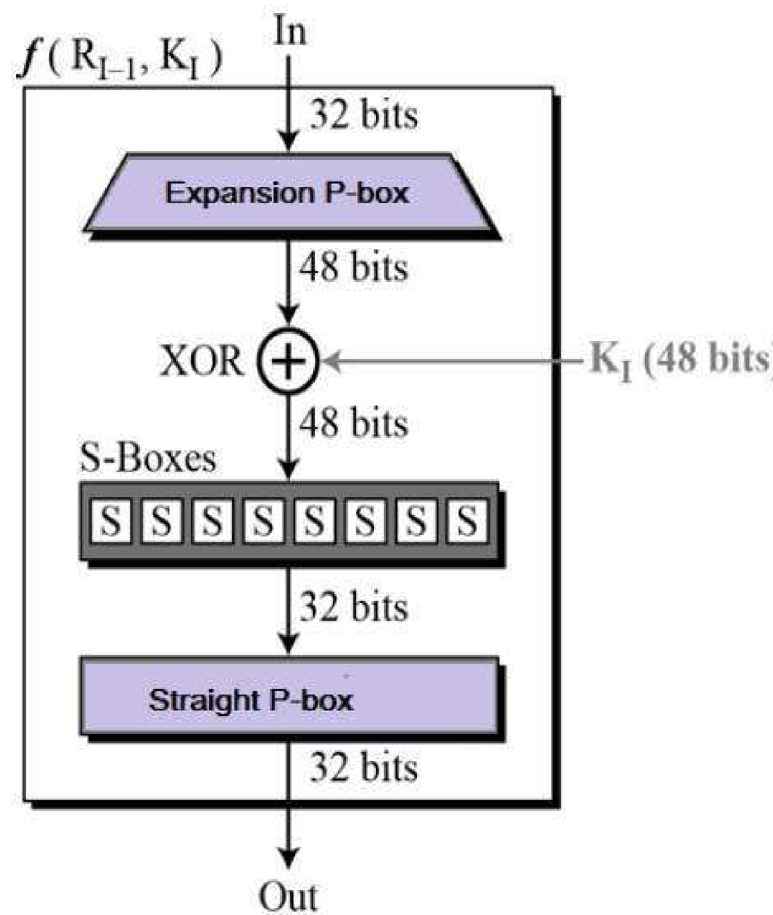
DES Round



32 bits
32 bits
28 bits
28 bits
L_{i-1}
R_{i-1}
C_{i-1}
D_{i-1}
Expansion Permutation (E table)
Left Shift(s)
Left Shift(s)
48 bits
Permutation/ Contraction (Permuted Choice 2)
48 bits
Substitution/ Choice (S-box)
32 bits
Permutation (P)
32 bits
32 bits
L_i
R_i
C_i
D_i
32 bits
32 bits



URE 10-11 Details of a Cycle.



DES' f function

- four stages
  - expand 32 bit half to 8 x 6-bit blocks (48 bits total)
  - XOR with round subkey
  - pass each 6-bit block through an S-box
    - reduces back to 32 bits
  - permute the bits to promote avalanche

Types of Permutations

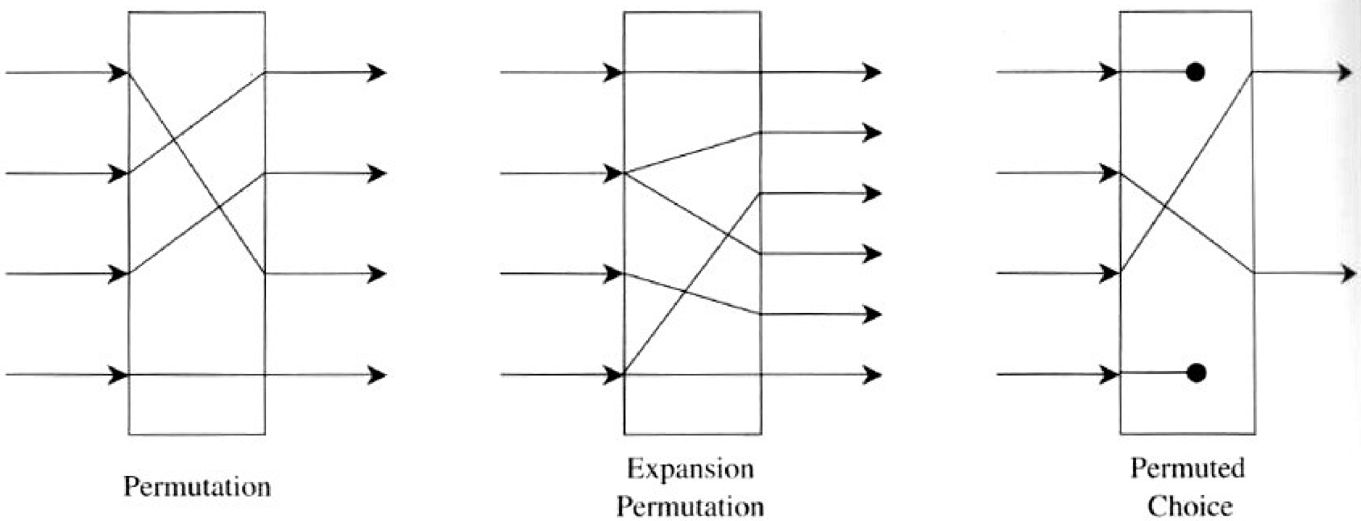
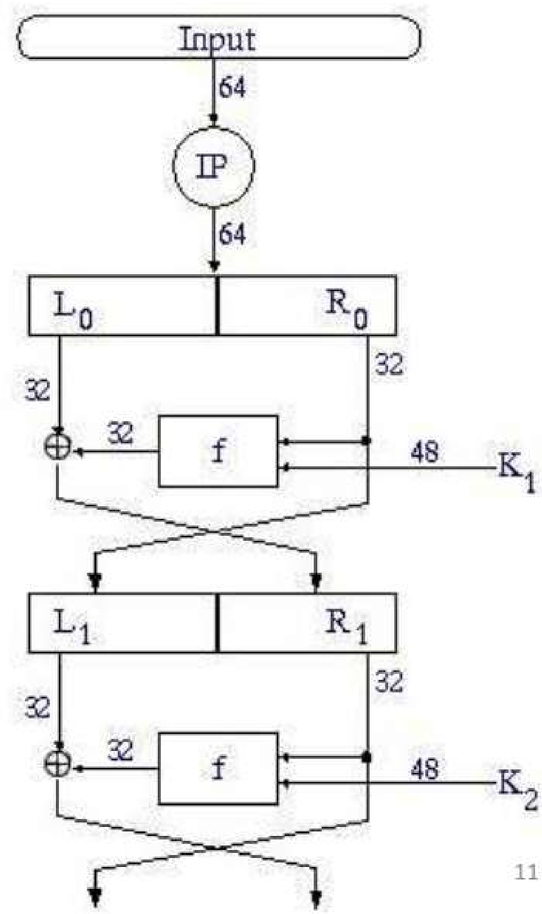


FIGURE 10-10 Types of Permutations.

- $IP(x) = L_0R_0$
- $L_i = R_{i-1}$
- $R_i = L_{i-1} \oplus f(R_{i-1}, K_i)$
- $y = IP^{-1}(R_{16}L_{16})$

Note: IP means Initial Permutation



How DES Works in Detail

Let M be the plain text message $M = 0123456789ABCDEF$, where M is in hexadecimal (base 16) format. Rewriting M in binary format, we get the 64-bit block of text:

- M = 0000 0001 0010 0011 0100 0101 0110 0111 1000 1001 1010 1011 1100 1101 1110 1111
- L = 0000 0001 0010 0011 0100 0101 0110 0111
- R = 1000 1001 1010 1011 1100 1101 1110 1111

- K = 133457799BBCDFF1
- K = 00010011 00110100 01010111 01111001 10011011 10111100 11011111 11110001

Step 1: Create 16 subkeys, each of which is 48-bits long

57	49	41	33	25	17	9
1	58	50	42	34	26	18
10	2	59	51	43	35	27
19	11	3	60	52	44	36
63	55	47	39	31	23	15
7	62	54	46	38	30	22
14	6	61	53	45	37	29
21	13	5	28	20	12	4

- K (64bit) =
- 00010011 00110100 01010111 01111001 10011011 10111100 11011111 11110001

- After permutation 56-bit using Table PC1
- K+ = 1111000 0110011 0010101 0101111 0101010 1011001 1001111 0001111

- Next, split this key into left and right halves, C0 and D0, where each half has 28 bits.
- C0 = 1111000 0110011 0010101 0101111
- D0 = 0101010 1011001 1001111 0001111

create sixteen blocks Ci and $Di, 1 <= i <= 16$

i	shift	Ci (28 bit)	Di (28 bit)
0		1111000011001100101010101111	0101010101100110011110001111
1	1	1110000110011001010101011111	1010101011001100111100011110
2	1	1100001100110010101010111111	0101010110011001111000111101
3	2	0000110011001010101011111111	0101011001100111100011110101
4	2	0011001100101010101111111100	0101100110011110001111010101
5	2	1100110010101010111111110000	0110011001111000111101010101
6	2	0011001010101011111111000011	1001100111100011110101010101
7	2	1100101010101111111100001100	0110011110001111010101010110
8	2	0010101010111111110000110011	1001111000111101010101011001
9	1	0101010101111111100001100110	0011110001111010101010110011
10	2	0101010111111110000110011001	1111000111101010101011001100
11	2	0101011111111000011001100101	1100011110101010101100110011
12	2	0101111111100001100110010101	0001111010101010110011001111
13	2	0111111110000110011001010101	0111101010101011001100111100
14	2	1111111000011001100101010101	1110101010101100110011110001
15	2	1111100001100110010101010111	1010101010110011001111000111
16	1	1111000011001100101010101111	0101010101100110011110001111

Round Key Generation

Generating K1 (48bit) from C1D1 (28+28=56 bit)

- C1D1 = 1110000 1100110 0101010 1011111 1010101 0110011 0011110 0011110

- K1 = 000110 110000 001011 101111 111111 000111 000001 110010

PC2

14	17	11	24	1	5
3	28	15	6	21	10
23	19	12	4	26	8
16	7	27	20	13	2
41	52	31	37	47	55
30	40	51	45	33	48
44	49	39	56	34	53
46	42	50	36	29	32

TABLE 10-3 Choice Permutation to Select 48 Key Bits.

Key Bit	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Selected for Position	5	24	7	16	6	10	20	18	—	12	3	15	23	1
Key Bit	15	16	17	18	19	20	21	22	23	24	25	26	27	28
Selected for Position	9	19	2	—	14	22	11	—	13	4	—	17	21	8
Key Bit	29	30	31	32	33	34	35	36	37	38	39	40	41	42
Selected for Position	47	31	27	48	35	41	—	46	28	—	39	32	25	44
Key Bit	43	44	45	46	47	48	49	50	51	52	53	54	55	56
Selected for Position	—	37	34	43	29	36	38	45	33	26	42	—	30	40

Applying the initial permutation to M

Initial Permutation (64-bit block)

- **M** = 0000 0001 0010 0011 0100 0101 0110 0111 1000 1001 1010 1011 1100 1101 1110 1111
- **IP** = 1100 1100 0000 0000 1100 1100 1111 1111 1111 0000 1010 1010 1111 0000 1010 1010

- L0 = 1100 1100 0000 0000 1100 1100 1111 1111
- R0 = 1111 0000 1010 1010 1111 0000 1010 1010

IP

58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

Function f (Round 1)

L0 and R0 (32 bit each) and Kn 48 bits

- K1 = 000110 110000 001011 101111 111111 000111 000001 110010
- L1 = R0 = 1111 0000 1010 1010 1111 0000 1010 1010
- $R1 = L0 + f(R0, K1)$

For nth iteration

- Ln = Rn-1
- Rn = Ln-1 + f(Rn-1,Kn)

Function f: expand each block Rn-1

Expansion (from 32 bits to 48 bits)

- R0 = 1111 0000 1010 1010 1111 0000 1010 1010
- E(R0) = 011110 100001 010101 010101 011110 100001 010101 010101

E BIT-SELECTION TABLE

32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

Expansion Permutation in DES

TABLE 10-1 Expansion Permutation.

Bit	1	2	3	4	5	6	7	8
Moves to Position	2,48	3	4	5,7	6,8	9	10	11,13
Bit	9	10	11	12	13	14	15	16
Moves to Position	12,14	15	16	17,19	18,20	21	22	23,25
Bit	17	18	19	20	21	22	23	24
Moves to Position	24,26	27	28	29,31	30,32	33	34	35,37
Bit	25	26	27	28	29	30	31	32
Moves to Position	36,38	39	40	41,43	42,44	45	46	47,1

Function f: XOR E(Rn-1) with the key Kn

- K1 = 000110 110000 001011 101111 111111 000111 000001 110010
- E(R0) = 011110 100001 010101 010101 011110 100001 010101 010101
- K1+E(R0) = 011000 010001 011110 111010 100001 100110 010100 100111.

Function f: S boxes

- Each group of six bits (B_i) of Ri will give us an address in a different S box.
- Kn + E(Rn-1) =B1B2B3B4B5B6B7B8
- that address will be a 4 bit number ($S_i(B_i)$)
- $S_1(B_1)S_2(B_2)S_3(B_3)S_4(B_4)S_5(B_5)$ $S_6(B_6)S_7(B_7)S_8(B_8)$

S BOX S1

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

The first and last bits of ***B*** represent in base 2 a number in the decimal range 0 to 3

The middle 4 bits of B represent in base 2 a number in the decimal range 0 to 15

8 s-boxes

S1															
14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13
S2															
15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9
S3															
10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12
S4															
7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14
S5															
2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3
S6															
12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13
S7															
4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12
S8															
13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

TABLE 10-4 S-Boxes of DES.

Box	Row	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
S_1																	
	0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
	1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
	2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
	3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13
S_2																	
	0	15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
	1	3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
	2	0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
	3	13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9
S_3																	
	0	10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
	1	13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
	2	13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
	3	1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12
S_4																	
	0	7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
	1	13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
	2	10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
	3	3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14
S_5																	
	0	2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
	1	14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
	2	4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
	3	11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3
S_6																	
	0	12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
	1	10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
	2	9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
	3	4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13
S_7																	
	0	4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
	1	13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
	2	1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
	3	6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12
S_8																	
	0	13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
	1	1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
	2	7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
	3	2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

Function f: S boxes

- K1 + E(R0) = 011000 010001 011110 111010 100001 100110 010100 100111.

- S1(B1)S2(B2)S3(B3)S4(B4)S5(B5)S6(B6)S7(B7)S8(B8) = 0101 1100 1000 0010 1011 0101 1001 0111

Function f: permutation

Final Permutation

- permutation P of the S-box output to obtain the final value of f
- $f = \mathbf{P}(S_1(B_1)S_2(B_2)\ldots S_8(B_8))$
- f = 0010 0011 0100 1010 1010 1001 1011 1011

Permutation table P

16	7	20	21
29	12	28	17
1	15	23	26
5	18	31	10
2	8	24	14
32	27	3	9
19	13	30	6
22	11	4	25

Finally

- R1 = L0 + f(R0 , K1)

= 1100 1100 0000 0000 1100 1100 1111 1111

+ 0010 0011 0100 1010 1010 1001 1011 1011

= 1110 1111 0100 1010 0110 0101 0100 0100

$\boldsymbol{L_2 = R_1}$

$\boldsymbol{R_2 = L_1 + f(R_1, K_2)}$, and so on for 16 rounds.

Try

- plaintext message "8787878787878787“
- key "0E329232EA6D0D73“

- Then
- ciphertext "0000000000000000"

TABLE 10-5 Permutation Box P.

Bit	Goes to Position							
1–8	9	17	23	31	13	28	2	18
9–16	24	16	30	6	26	20	10	1
17–24	8	14	25	3	4	29	11	19
25–32	32	12	22	7	5	27	15	21

TABLE 10-6 Initial Permutation.

Bit	Goes to Position							
1–8	40	8	48	16	56	24	64	32
9–16	39	7	47	15	55	23	63	31
17–24	38	6	46	14	54	22	62	30
25–32	37	5	45	13	53	21	61	29
33–40	36	4	44	12	52	20	60	28
41–48	35	3	43	11	51	19	59	27
49–56	34	2	42	10	50	18	58	26
57–64	33	1	41	9	49	17	57	25

Attacking Block Ciphers

- Types of attacks to consider
  - known plaintext given several pairs of plaintexts and ciphertexts, recover the key (or decrypt another block encrypted under the same key)
  - how would chosen plaintext and chosen ciphertext be defined?
- Standard attacks
  - exhaustive key search
  - dictionary attack
  - differential cryptanalysis, linear cryptanalysis
- Side channel attacks.

DES's main vulnerability is short key size.

Chosen-Plaintext Dictionary Attacks Against Block Ciphers

- Construct a table with the following entries
  - (K, $E_K[0]$) for all possible key K
  - Sort based on the second field (ciphertext)
  - How much time does this take?
- To attack a new key K (under chosen message attacks)
  - Choose 0, obtain the ciphertext C, looks up in the table, and finds the corresponding key
  - How much time does this step take?
- Trade off space for time

References

- William Stallings, Network Security Essentials : Applications and Standards, ISBN: 9788131761755, 8131761754

- Thanks to the many unknown sources from where some information is adopted.